This document will be where I take notes for myself and the development as I work through this project.

Project1: Following along with learnopengl.com

## 1/22/22: Not all that Glitters is gold.

“Glitter compiles and statically links every required library, so you can jump right into doing what you probably want: how to get started with OpenGL.”

The [Glitter project](https://github.com/Polytonic/Glitter) allegedly allows me to make a new project easily, but it looks like I might need to modify it somewhat to actually make it run. So much for no setup, at least it has the ‘vendor’ dependencies like GLFW included.  
Make Deprecation Warning at Glitter/Vendor/bullet/CMakeLists.txt:1 (cmake\_minimum\_required):

 Compatibility with CMake < 2.8.12 will be removed from a future version of

 CMake.

 Update the VERSION argument <min> value or use a ...<max> suffix to tell

 CMake that the project does not need compatibility with older versions.

<https://cmake.org/cmake/help/latest/command/cmake_policy.html> ultimately would have to implement for each individual warning

<https://www.foonathan.net/2018/10/cmake-warnings/> This was helpful.

In the end I had to experiment with it and determined that include\_directories(SYSTEM "${LIB\_DIR}/Glitter/Vendor") was the best way to go. This was put right after the if (MSVC) ... endif(). I am pretty sure that include\_directories(SYSTEM "${LIB\_DIR}/Glitter") would suppress warning errors for the entire project. Since I want my headers, sources, and shader folders to be checked for warnings.

Then, I set up a git repo. The .gitignore file has had the build and vendor folders ignored.

Running cmake .., then make, I have an error within the headers. Okay, now what? It’s assimp, the mesh loading library.

**/home/royal/sourcecode/cpp/learnopengl/learnopengl\_project1/Glitter/Vendor/assimp/include/assimp/defs.h:55:10:** **fatal error:** assimp/confi

g.h: No such file or directory

  55 | #include **<assimp/config.h>**

I’m not here to fix assimp’s make file problems, so I’ll just get rid of assimp and reinstall it when I need to. This will need me to remove it from the include\_directories in the CMakeLists.txt. I’ll also need to remove it from glitter.hpp.

In fact, the learnopengl page for installing assimp says to compile it yourself to avoid errors.

Maybe I should just set up my own cmake stuff at this point. I still learned a lot and refreshed some cmake stuff in the process.

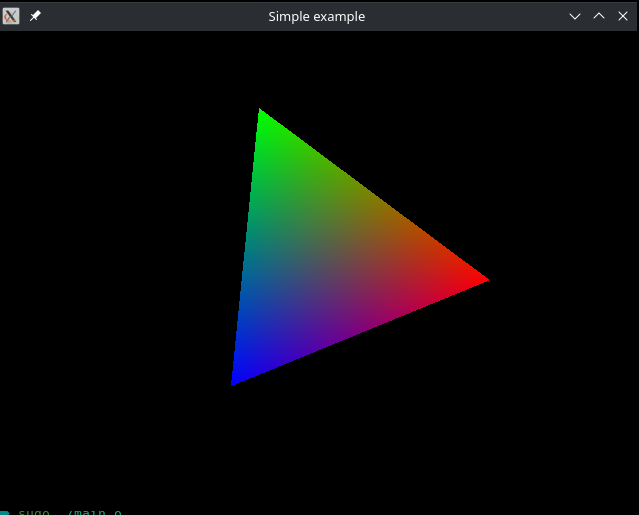
## 1/23/22: Starting over from not-scratch.

So, the tutorial from opengltutorial.com, which I’ve previously been following, got me up and running and even displaying a 3D cube. So I’m going to start there, which is not too far ahead of starting from the beginning of learnopengl.

[https://learnopengl.com/Getting-started/Hello-Window](https://learnopengl.com/Getting-started/Hello-Triangle) Good place to start.

## 1/25/22: Starting over from scratch.

Just following the learnopengl tutorial from the beginning, including compiling glfw and stuff. The tutorial uses visual studio but I had to consult this [stack overflow page.](https://stackoverflow.com/questions/17768008/how-to-build-install-glfw-3-and-use-it-in-a-linux-project)



Time to make my own cmake file

Rather, steal the test one? Where is that? A job for next time. Might need to go back to the GLFW example tutorial and find what

<https://www.glfw.org/docs/3.0/quick.html> says to do cuz that worked

## 1/30/22: Cmake

Todo: kate build plugin

<https://www.glfw.org/docs/latest/build_guide.html> “With Cmake and GLFW source”

<https://github.com/juliettef/GLFW-CMake-starter/blob/main/CMakeLists.txt>

<https://schneide.blog/2016/04/08/modern-cmake-with-target_link_libraries/>   
  
Finally got glfw and glad headers to compile without error. That makes me ready to start Getting Started > Hello Window.

<https://www.glfw.org/docs/latest/group__keys.html>

## 2/2/22: glad

<https://github.com/Dav1dde/glad/issues/251>

For future reference, this <https://github.com/juliettef/GLFW-CMake-starter> is a lot easier to start from. It just works!

